2023

CNN을 이용한 제품 구분

작성일 : 2023.06.08
학번 : 2023254002
성명 : 이민수

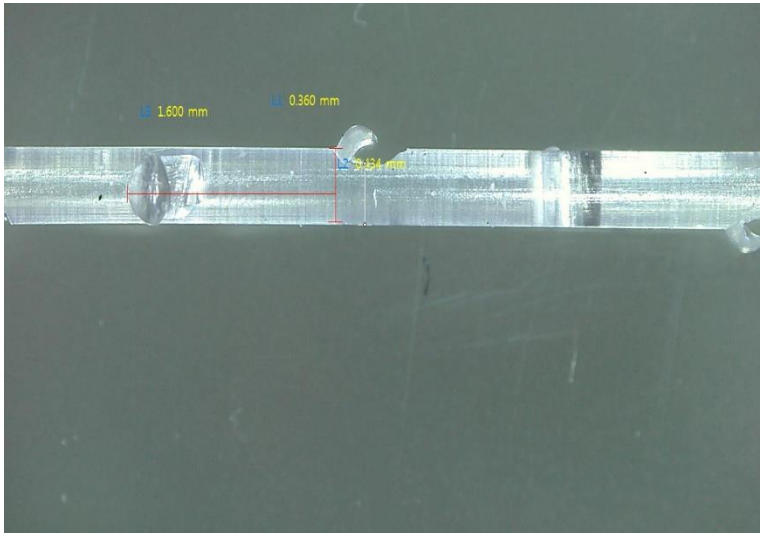
01 개요

CNN을 이용한 제품 구분

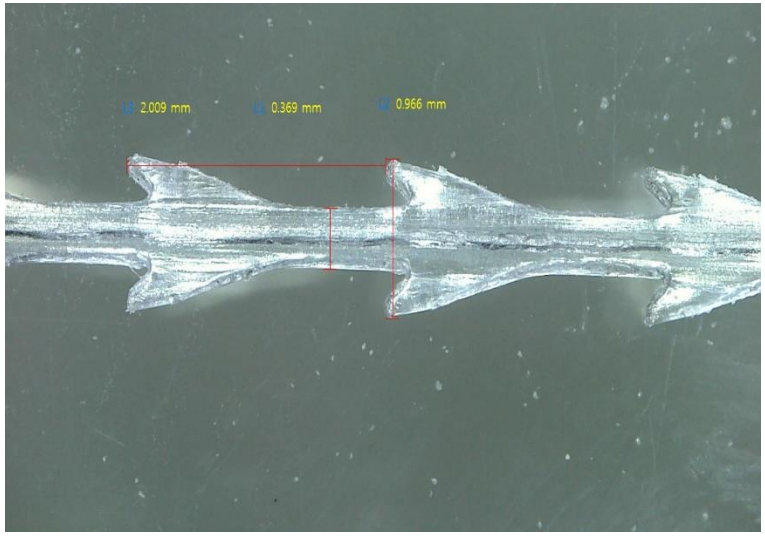
1. 개요

Sample Name	Double	Kog
Sample 개수	test 50EA train 50EA	test 50EA train 50EA
사용 Model	Relu 및 Sigmoid	

2. Sample 사진



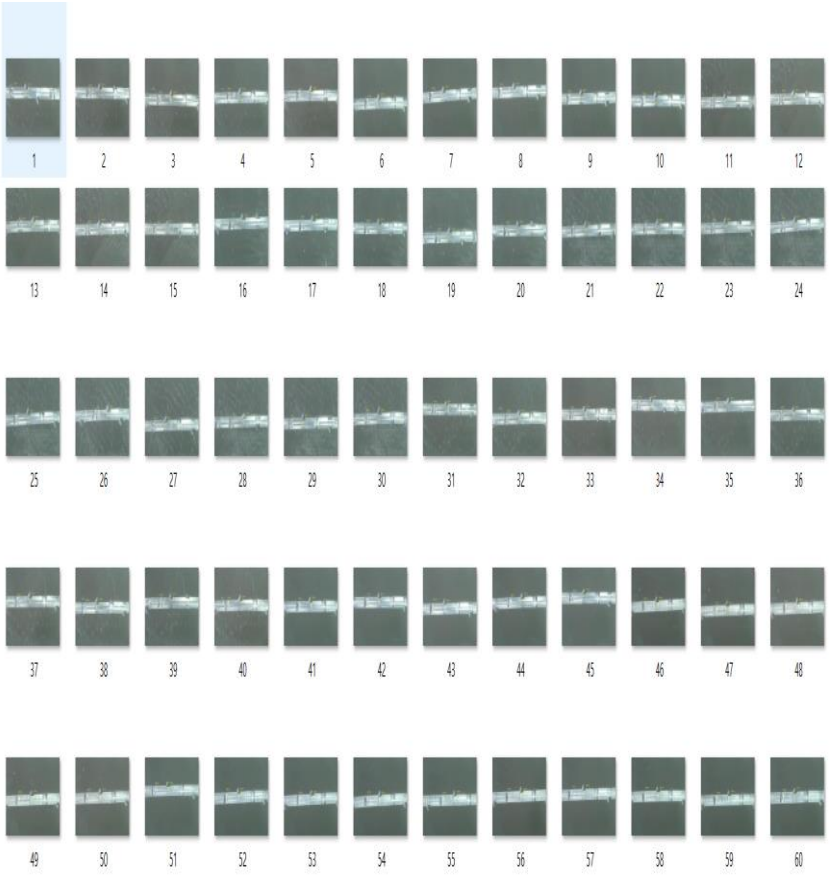
Double



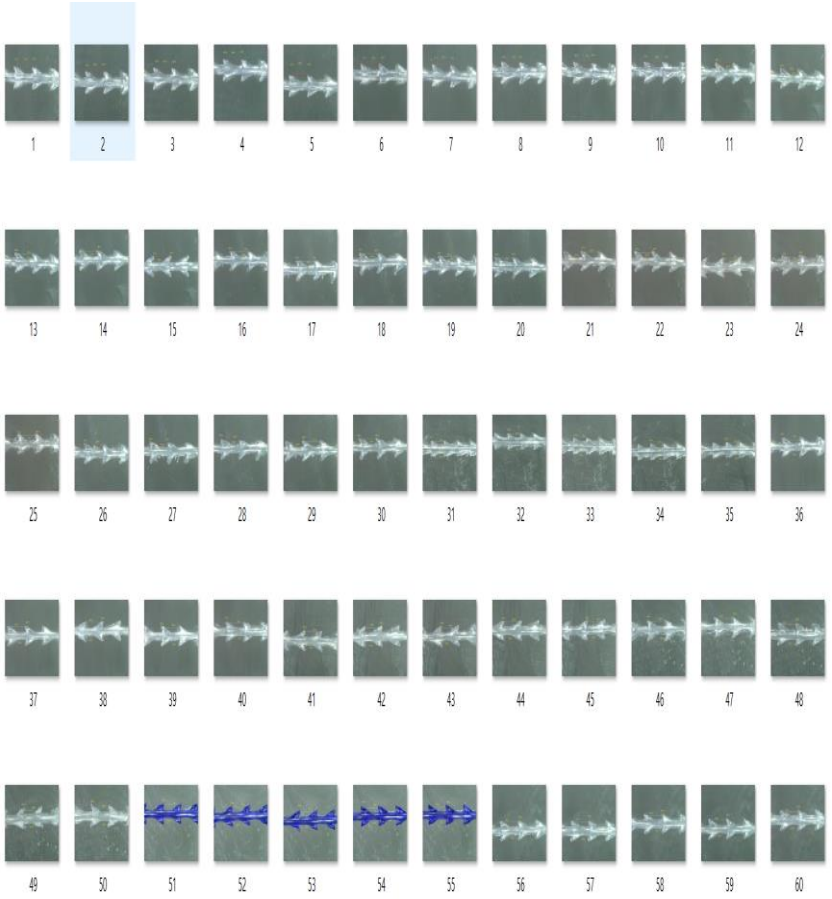
Kog

01 개요

CNN을 이용한 제품 구분



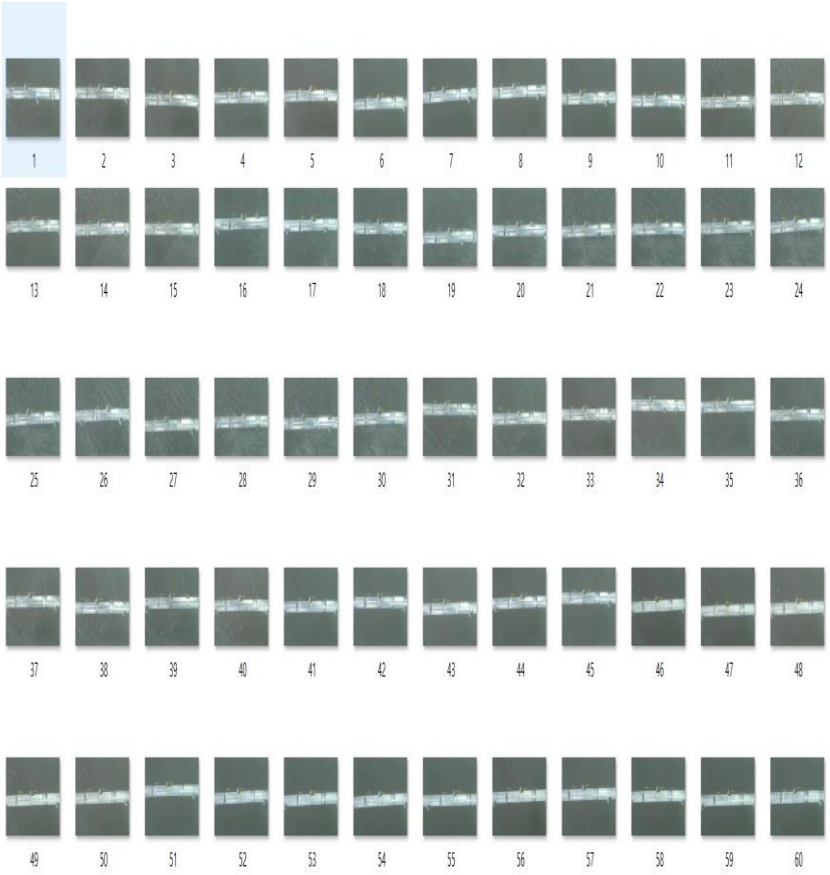
Double



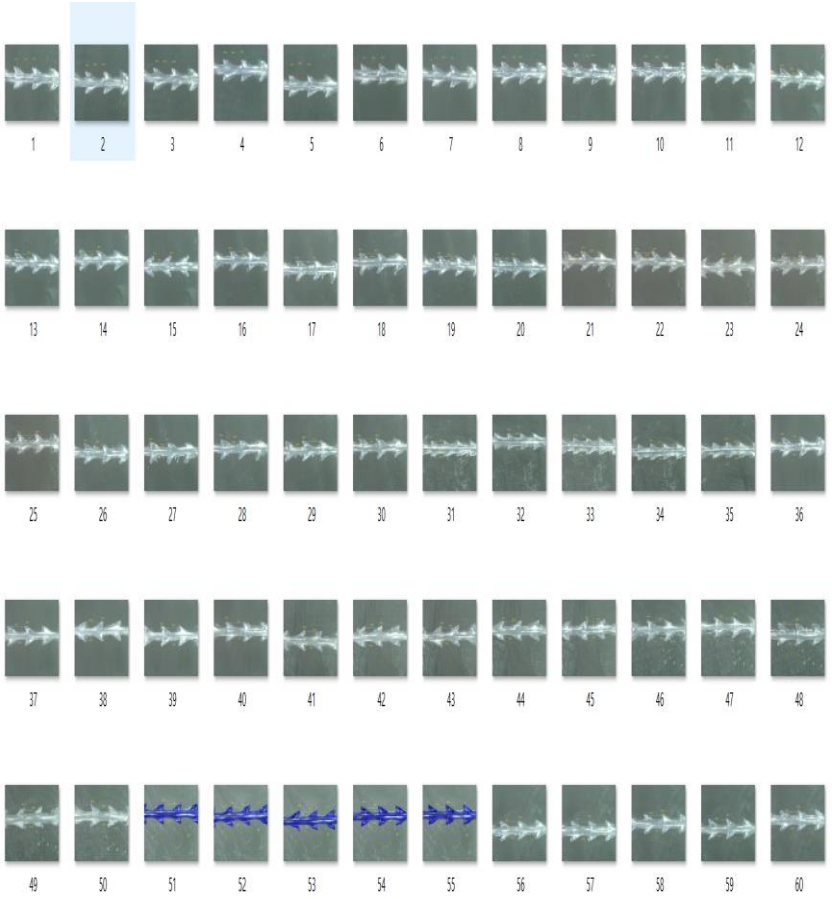
Kog

01 개요

CNN을 이용한 제품 구분



Double



Kog

01 개요

CNN을 이용한 제품 구분

```
import os
import numpy as np
import tensorflow as tf
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense

# Mount Google Drive to access the dataset
from google.colab import drive
drive.mount('/content/drive')

# Path to the folders containing the images
Double_path = '/content/drive/MyDrive/Double'
Kog_path = '/content/drive/MyDrive/Kog'

# Parameters
image_size = (64, 64)
batch_size = 32

# Data augmentation and preprocessing
train_datagen = ImageDataGenerator(
    rescale=1.0/255.0,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True
)
```

01 개요

CNN을 이용한 제품 구분

```
test_datagen = ImageDataGenerator(rescale=1.0/255.0)

# Load and preprocess the training set
train_set = train_datagen.flow_from_directory(
    Double_path,
    target_size=image_size,
    batch_size=batch_size,
    class_mode='binary'
)

# Load and preprocess the test set
test_set = test_datagen.flow_from_directory(
    Kog_path,
    target_size=image_size,
    batch_size=batch_size,
    class_mode='binary'
)

# Create the CNN model
model = Sequential()
model.add(Conv2D(32, (3, 3), activation='relu', input_shape=(image_size[0], image_size[1], 3)))
model.add(MaxPooling2D((2, 2)))
model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(MaxPooling2D((2, 2)))
model.add(Flatten())
model.add(Dense(64, activation='relu'))
model.add(Dense(1, activation='sigmoid'))

# Compile the model
model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])
```

01 개요

CNN을 이용한 제품 구분

```
# Train the model
model.fit(
    train_set,
    steps_per_epoch=train_set.samples // batch_size,
    epochs=100,
    validation_data=test_set,
    validation_steps=test_set.samples // batch_size
)
```

02 결론

CNN을 이용한 제품 구분

```
3/3 [==============================] - 3s 1s/step - loss: 0.6453 - accuracy: 0.6324 - val_loss: 0.6915 - val_accuracy: 0.5417
Epoch 88/100
3/3 [==============================] - 3s 1s/step - loss: 0.6355 - accuracy: 0.5588 - val_loss: 0.6924 - val_accuracy: 0.5417
Epoch 89/100
3/3 [==============================] - 4s 2s/step - loss: 0.5983 - accuracy: 0.6765 - val_loss: 0.6800 - val_accuracy: 0.5729
Epoch 90/100
3/3 [==============================] - 3s 1s/step - loss: 0.6278 - accuracy: 0.6146 - val_loss: 0.6904 - val_accuracy: 0.5521
Epoch 91/100
3/3 [==============================] - 3s 1s/step - loss: 0.6310 - accuracy: 0.6029 - val_loss: 0.6912 - val_accuracy: 0.5312
Epoch 92/100
3/3 [==============================] - 3s 2s/step - loss: 0.6243 - accuracy: 0.6176 - val_loss: 0.6875 - val_accuracy: 0.5521
Epoch 93/100
3/3 [==============================] - 3s 1s/step - loss: 0.6346 - accuracy: 0.5735 - val_loss: 0.7265 - val_accuracy: 0.4792
Epoch 94/100
3/3 [==============================] - 4s 2s/step - loss: 0.6458 - accuracy: 0.5882 - val_loss: 0.7531 - val_accuracy: 0.4792
Epoch 95/100
3/3 [==============================] - 3s 2s/step - loss: 0.6529 - accuracy: 0.5735 - val_loss: 0.7354 - val_accuracy: 0.4583
Epoch 96/100
3/3 [==============================] - 4s 2s/step - loss: 0.6397 - accuracy: 0.6029 - val_loss: 0.7264 - val_accuracy: 0.4375
Epoch 97/100
3/3 [==============================] - 3s 1s/step - loss: 0.6422 - accuracy: 0.5625 - val_loss: 0.7288 - val_accuracy: 0.4792
Epoch 98/100
3/3 [==============================] - 4s 2s/step - loss: 0.6621 - accuracy: 0.5441 - val_loss: 0.7189 - val_accuracy: 0.5000
Epoch 99/100
3/3 [==============================] - 4s 1s/step - loss: 0.6047 - accuracy: 0.7353 - val_loss: 0.7244 - val_accuracy: 0.4792
Epoch 100/100
3/3 [==============================] - 3s 1s/step - loss: 0.6307 - accuracy: 0.6354 - val_loss: 0.7077 - val_accuracy: 0.5104
<keras.callbacks.History at 0x7f50be7eec50>
```

02 결론

CNN을 이용한 제품 구분

3. 결론

- **Loss 0.6307 / accuracy 0.6354를 확인 하였음.**
- **정확도 및 오차가 신뢰할 수 있는 수준이 못 미친다고 판단 함. 추가적인 Sample 이미지 개수 추가 및 code의 수정이 필요하다고 생각됨.**